Journal of the Korean Institute of Industrial Engineers
Vol. 48, No. 4, pp. 355-366, August 2022.
ISSN 1225-0988 | EISSN 2234-6457

https://doi.org/10.7232/JKIIE.2022.48.4.355

Two Phase Multivariate to Multivariate Time Series Forecasting Using Self-attention Convolutional Autoencoder and Temporal Convolutional Network

Woo Young Hwang · Jun-Geol Baek†

Department of Industrial and Management Engineering, Korea University

Self-attention Convolutional Autoencoder와 Temporal Convolutional Network를 이용한 Two Phase 다변량 시계열 예측

황우영 · 백준걸

고려대학교 산업경영공학과

In manufacturing process, data is collected in the form of correlated sequences. Multivariate to multivariate time series (MMTS) forecasting is an important factor in manufacturing. MMTS forecasting is a notoriously challenging task considering the need for incorporating both non-linear correlations between variables (inter-relationships) and temporal relationships of each univariate time series (intra-relationships) while forecasting future time steps of each univariate time series (UTS) simultaneously. However, previous works use deep learning models suited for low-dimensional data. These models are insufficient to model high-dimensional relationships inherent in multivariate time series (MTS) data. Furthermore, these models are less productive and efficient as they focus on predicting a single target variable from multiple input variables. Thus, we proposed two phase MTS forecasting. First, the proposed method learns the non-linear correlations between UTS (inter-relationship) through self-attention based convolutional autoencoder and conducts cause analysis. Second, it learns the temporal relationships (intra-relationships) of MTS data through temporal convolutional network and forecasts multiple target outputs. As an end-to-end model, the proposed method is more efficient and derives excellent experimental results.

Keywords: Cause Analysis, Inter-Relationship, Intra-Relationship, Multivariate to Multivariate Time Series Forecasting, Self-Attention Convolutional Autoencoder, Temporal Convolutional Network

1. Introduction

Time series forecasting is a vital component in many industrial applications such as prognostics and health management, optimization, dynamic scheduling, and quality control (Morariu *et al.*, 2018; Mawson *et al.*, 2020). In current manufacturing domains, the increasing availability of time series data and machinery data provides abundant information. This is because, millions of multivariate time

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2022R1A2C2004457). Also, this work was supported by Brain Korea 21 FOUR and Samsung Electronics Co., Ltd (IO201210-07929-01).

† Corresponding author : Professor Jun-Geol Baek, Department of Industrial and Management Engineering, Korea University, 145 Anam-ro, Seongbuk-gu, Seoul, 02841, Korea, Tel : +82-2-3290-3396, Fax : +82-2-3290-4550, E-mail : jungeol@korea.ac.kr

Received February 3, 2022; Revision Received March 30, 2022; Accepted May 23, 2022.

series (MTS) data are produced by the minute via different sensors embedded in industrial machineries and in-between manufacturing lines. As such, the demand for MTS forecasting has risen in the manufacturing domain. For example, an engineer may schedule a timely maintenance event by forecasting machine MTS sensor outputs and predicting machinery remaining useful life (RUL). With the basic assumption that individual variables of MTS data are dependent on each other, exploiting the non-linear dependencies between the variables and forecasting accurate future time steps has become a key factor to success in the highly competitive industry.

Time series forecasting typically uses certain amount of time series data to construct a model that can forecast future outputs based on training data. To forecast accurate outputs, adequate representative features of the training data must be well captured by the model.

For univariate time series (UTS) forecasting as general, two key components must be considered. As aforementioned, it is important to learn the characteristics of each time series data, stationary or non-stationary, has a trend, seasonality and so on (Braei and Wagner, 2020). Solving the long-term dependency problem also remains a chronic challenge of forecasting accurate future time steps. The ability to leverage the long-term dependencies between the current time step and the past time steps is critical to the prediction capacity. However, for MTS forecasting with high dimensional settings, addition to the prior components, the complex distribution of the input series and the non-linear correlations between the variables must be considered as well (Wu *et al.*, 2020).

When conducting time series forecasting, there are three main methods that are extensively utilized, statistical methods, machine learning methods, and deep learning methods. Traditional time series forecasting methods for univariate statistical methods are moving average (MA), autoregressive (AR), autoregressive integrated moving average (ARIMA), and exponential smoothing (ES) (Hyndman and Athanasopoulos, 2018). Multivariate statistical methods are vector autoregressive (VAR) model and vector autoregressive moving average (VARMA). These models exhibit strong explanatory characteristics. They are not, however, sufficient to model the relationships of large, complex, and high-dimensional data as these methods operate on linear state-spaces.

Machine learning forecasting models are random forest (RF) and regression models such as Support Vector Regressor (SVR). These have shown relative weakness in forecasting accuracy compared to simple statistical models when train data is insufficient (Cerqueira *et al.*, 2019).

Deep learning methods are extensively utilized to solve multiple real-life problems in manufacturing domains and made exceptional impacts along the way. Recurrent Neural Network (RNN) models have been used to forecast occurrence of process failures. As examples, Meyes *et al.* (2019) have trained two Long Short-Term Memory (LSTM) networks with the auto-labeled data to predict process failures. Wang *et al.* (2019) have conducted predictive analytics in smart manufacturing using Gated Recurrent Unit (GRU) models (Chung *et al.*, 2014). However, although LSTM and GRU are known to alleviate vanishing/exploding problems in sequence modeling tasks, studies have shown that gradient norms still decay exponentially fast with delay (DiPietro and Hager, 2020).

In the proposed method, to substitute the aforementioned models, temporal convolutional network (TCN) is proposed. TCN is comprised of dilated convolutions and causal convolutions and was advanced by the wavenet model (Oord *et al.*, 2016). TCN exhibits few advantageous characteristics such as parallelism, flexible receptive field size, and stable gradients. The model has recently been used widely in sequence modeling and proved to outperform baseline recurrent architectures on a broad range of sequence modeling tasks (Bai *et al.*, 2018).

In this paper, we term the concept of the non-linearly correlated relationships between UTS data in MTS as inter-relationship and the temporal dependencies between the time domains within each UTS as intra-relationship.

Manufacturing MTS data, which encompasses inter-relationships and intra-relationships, is innately high-dimensional in comparison to UTS data. As an example, vibration of a machinery may be affected by relative components of the machine with non-linear correlations (i.e., sound frequency, pressure, rotation, voltage and so on). For forecasting manufacturing MTS data, inter-relationships are leveraged to capture the dependencies between multiple variables. Also, intra-relationships are leveraged to capture the temporal dependencies of the past and the current time step of individual univariate data. Current existing forecasting models which are oriented toward UTS forecasting lack the ability to capture both the non-linear relationships and the temporal relationships. These models lack the ability to benefit from the high-dimensional settings of MTS data. Furthermore, current multivariate models are focused on forecasting a single representative target variable. On the contrary, multivariate to multivariate time series (MMTS) forecasting can be more beneficial in manufacturing environments. As MMTS forecasts future values of corresponding multiple input values simultaneously and provides cause analysis, it provides users with more explicit details of future status of a machinery.

The input to Phase Ⅰ of the proposed method is a MTS data, $X^i = \{x_0^i, \cdots, x_t^i\}$, where $X^i \in R^{i \times t}$ is the time points of *i*-th variable in the MTS data. Input to Phase Ⅱ TCN module is

$X' = \{x_0', \cdots, x_t'\}$, where $X' \in R^t$, a condensed UTS representation of X^i in Phase Ⅰ. MMTS forecast task is to forecast the corresponding $\hat{y}^i_{t+a}$ from the given MTS input data, while the ground truth value is expressed as $X' = \{x_0', \cdots, x_t'\}$. The Equation (1) is presented as follows:

$$\hat{y}^i_{t+a} = x^i_{t+a} = f(x^i_0,\ \ldots,\ x^i_t;\theta) + \epsilon \tag{1}$$

$f(\cdot)$ is the model, a is the range of future time steps, θ learnable parameter, and ϵ prediction error of the model.

In this paper, we proposed two phase MMTS forecasting. The proposed method is a novel two phase MMTS forecasting model utilizing self-attention based convolutional autoencoder (SACAE) and temporal convolutional network (TCN). In summary, the contributions of this study are as follows.

- Phase Ⅰ of the proposed method captures the non-linear correlated relationships of MTS data with SACAE and converts to UTS which represents the non-linear correlation of MTS data.
- Phase Ⅱ captures the temporal pattern of representative UTS created by Phase Ⅰ. We enable MMTS forecasting by restoring the forecasted representative UTS to corresponding variables in MTS data by the trained decoder of SACAE.
- The proposed method is unique in that MTS is forecasted not into a single target value but to original multi variables.
- MMTS forecasting by the proposed method can provide detailed information, cause analysis, of the state of MTS.

The rest of the paper is structured as follows: In Section Ⅱ, we introduce the underlying modules and why these modules are used in the proposed method. We then address the overall architecture of the proposed method and the operational sequences of the proposed method in detail in Section Ⅲ. Finally, in Section Ⅳ, we report experiments with multiple datasets that can demonstrate how the proposed method exhibits advantages in MMTS forecasting compared to baseline sequence models.

2. Background

This section introduces the sub-modules used in the proposed method and the reason for implementing these modules, convolutional autoencoder (CAE), self-attention, and TCN.

2.1 Convolutional Autoencoder

CAE is a variant of neural network aimed at reconstruction and feature extraction. Convolutional Neural Network (CNN) can dynamically generate key features from input images due to its local receptive field, weights sharing and pooling characteristics (Zhao *et al.*, 2017). Autoencoder (AE) can transform feature vectors into abstract feature vectors that can learn inter-relationship from high dimensional data space to low dimensional data space (Yan and Han, 2018). In a previous study, Zheng *et al.* (2014) proposed feature learning technique for MTS data by separating MTS into UTS and using deep CNN to derive good classification results. This technique, however, has a major limitation that inter-relationship between UTS cannot be incorporated. To overcome this shortcoming, inspired by CNN and AE for their feature extraction characteristics, we explore these frameworks for MMTS forecasting. An aggregate of two models, CNN and AE, CAE extracts representative features of the local map of the given input. Different from the previous study, instead of separating MTS into UTS, we jointly use MTS for feature extraction. With weight sharing structural characteristics of CNN, CAE has fewer training parameters, higher training efficiency, and avoids over-fitting (Wu *et al.*, 2021).

In the proposed method, CAE is used to extract non-linear relationships of multiple variables of MTS data. Adding local convolutions with AE, CAE can encode the original input by extracting the local features of the input into a feature map. Then convolutional decoder decodes the extracted feature map into the original output. Due to the local feature capturing characteristics, multiple filters are used in the same convolutional layer of CAE to extract diverse features. To elaborate, MTS inter-relationship extraction is illustrated in <Figure 1>.

The universal convolutional operation of CAE can be expressed as Equation (2):

$$Z_c = h(W_c * X^i + b_c) \tag{2}$$

W_c and b_c are the weight parameters. $h(\cdot)$ is the activation function and * denotes dot product of the convolutional layer and the pixels. The *c*-th filter scans the input matrix X^i and produces Z_c, output of the convolutional operation.

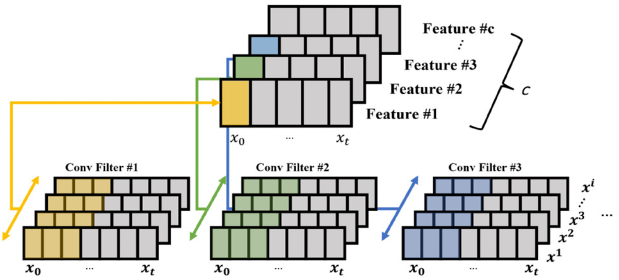

Figure 1. Architecture for MTS Inter-relationship Feature Extraction

After the convolutional operation, an activation function of Parametric Rectified Linear Unit (PReLU) is utilized. PReLU is an activation function that generalizes the traditional rectified unit with a slope for negative values. PReLU enables different types of nonlinearities for different layers and is expressed as Equation (3):

$$h(T_c) = \begin{cases} T_c, & \text{if } T_c > 0 \\ p_c T_c, & \text{if } T_c \le 0 \end{cases} \tag{3}$$

$T_c = W_c^* X^i + b_c$ is the c-th channel input to the activation function $h(\cdot)$ and p_c is the coefficient for the control of negative part of the activation function. If p_c becomes 0, the activation function becomes ReLU. When p_c is a learnable parameter, Equation (3) is PReLU (He *et al.*, 2015).

2.2 Self-attention

CNN and its local convolutional kernel limit the convolutional operation to capture only local information of the kernel (Wu *et al.*, 2021). <Figure 2> represents the convolutional kernel of CNN. If the kernel size becomes 1 x 3, the result is only dependent on the three values of the receptive field. To secure a wider information, convolutions are stacked several times, or the size of the data is reduced through the pooling layer. However, widening the receptive field is not efficient as the depth of the network must become deep enough to secure a sufficient size of the receptive fields. This method is not a sufficient solve to the non-local dependency problem.

Wang *et al.*(2018) proposed a non-local neural network, a type of self-attention module, that is efficient and can be well integrated into the existing deep learning architectures. By implementing the self-attention operations, the proposed model can capture the non-local dependencies by only the interactions between any two positions. Furthermore, analyzing the attention map created from the self-attention module enables cause analysis after MMTS forecasting. The self-attention module for the proposed method is depicted in <Figure 3>.

The residual style of the self-attention module output is expressed as Equation (4), shown below.

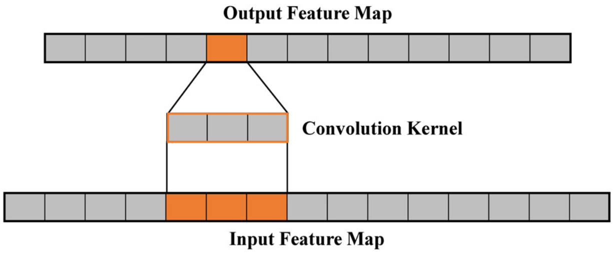

Figure 2. Figure of CNN Convolutional Kernel

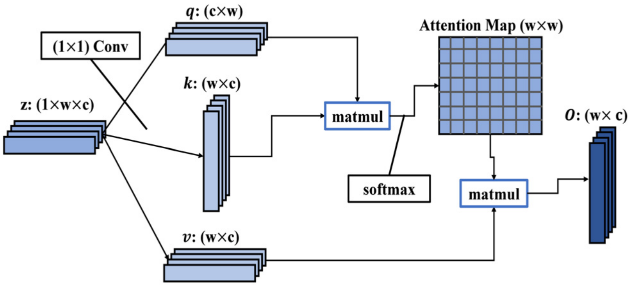

Figure 3. Self-attention in the Proposed Method, 1×1 Convolution is Applied to the Input to Create Query, Key, and Value. The Final Output is Added to the Input, Similar to the Residual Connection

$$X' = Z_c + O_c \tag{4}$$

The output feature map of the self-attention $O_c \in R^{1\times W\times C}$ is added to the original input $Z_c \in R^{1\times W\times C}$. 1, W, and C represent the height, the width, and the channel of the output of the convolutional operation and the output feature map. The residual style of the attention feature enables self-attention to be inserted in the last and the first layer of the convolutional layers of the encoder and the decoder of CAE. Self-attention module is calculated by the following Equation (5):

$$\Phi(z_i, z_j) = e^{k(z_j)^T \cdot q(z_i)} \tag{5}$$

z_i and z_j are feature maps of a certain convolutional layer and $k(\cdot)$, and $q(\cdot)$ are 1x1 convolutions. T transposes the variable. Applying 1×1 convolutions on the feature maps reduce the number of channels and computation cost (Wu *et al.*, 2021). The attention weights are calculated by normalizing the output score of Equation (5). The attention weights are calculated by following Equation (6):

$$\psi(i, j) = \frac{\Phi(z_i, z_j)}{\sum_{k=1}^{1\times w}\Phi(z_i, z_k)} \tag{6}$$

Next, the attention weight is calculated with the input feature map to produce the attention applied output feature map O_c. Equation of the output feature map is presented below in Equation (7):

$$O_c = \sum_{j=1}^{w} \Psi(z_i, z_j) \cdot v(z_j) \tag{7}$$

$v(\cdot)$ is 1×1 convolution.

Considering Equation (5) is computed based on the relationships between feature maps of the convolutional layer, the self-attention module can mitigate local dependency issues.

2.3 Temporal Convolutional Network

TCN is a variant of convolutional architectures for sequential data, which includes causal convolutions and dilated convolutions. In a previous study such as Bai *et al.* (2018), TCN architecture was experimented against traditional recurrent architectures across comprehensive sequence modeling tasks. In the previous study it was concluded that TCN outperforms generic LSTMs and GRUs.

For MMTS forecasting it is required to reproduce the original MTS input from the condensed UTS representation. Thus, an accurate forecast ability of the condensed UTS representation must be a foreground operation. However, LSTMs and GRUs exhibit vanishing/exploding gradient problems, which we deemed inappropriate for accurate forecasting of the condensed UTS representation.

TCN lists several advantages compared to these models. One of the major advantages of TCN is its ability to stabilize gradients. TCN can avoid vanishing/exploding gradients as it has a back-propagation path different from the temporal direction of the sequence (Bai *et al.*, 2018).

Before explaining the model structure of TCN, we define the nature of sequence modeling task of the condensed UTS representation, where the condensed UTS representation $X' = \{x_0', \ldots, x_t'\}$ is trained to predict the expected output of $x'_{t+1}, \ldots, x'_{t+a}$. a is the range of future time steps. The core of the sequence prediction lies in relying only on the given past input sequence of $x_0', \ldots, x_t'$ and not on the future input sequence of $x'_{t+1}, \ldots, x'_{t+a}$ to predict the output x'_{t+a}. The general function is as Equation (8):

$$x'_{t+a} = r(x_0', \ldots, x_t') \tag{8}$$

$r(\cdot)$ is TCN model.

Causal convolutions satisfy the aforementioned point where the nature of time series data, the future output at time $t+a$ must satisfy the ‘causality’ of relying on only the present and the past, not the future data points. TCN uses 1-D convolution with zero-padding with ‘kernel size -1’ to achieve convolution only with elements from time t and earlier in the previous layer (Bai *et al.*, 2018). However, 1-D convolutions and causal convolutions require many layers and large filters to capture distant time events. To solve the inefficiency of requiring many layers and large filters to capture the distant time events, TCN utilizes dilated convolutions. Advanced by the wavenet, dilated convolutions increase the receptive field by the order of magnitude with low computation cost (Oord *et al.*, 2016). Dilation filter size is as Equation (9):

$$\textit{Dilation filter} = (p-1)\Sigma_l d_l + 1 \tag{9}$$

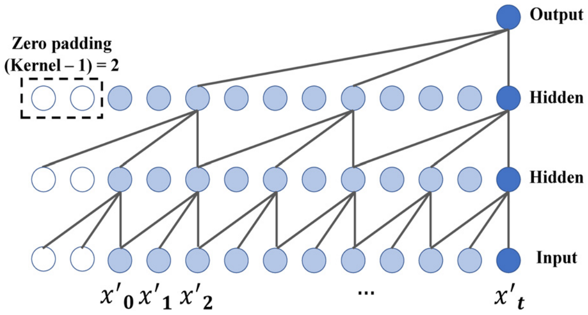

Figure 4. Basic Structural Explanation of TCN

Dilation size of layer l is d_l, filter size is p, and dilation rate is d. Inserting zeros in the convolutional kernels, convolution is applied with $d-1$ skipped inputs. The architecture of TCN, causal convolution and dilated convolution, is shown in <Figure 4>. By stacking several dilated convolution layers, TCN can achieve larger receptive fields with fewer layers compared to original convolution operations.

3. Proposed Method

In this section, we discuss the proposed method and the training method in further detail. Main structure of the proposed method is a two phase network, with Phase Ⅰ including SACAE and Phase Ⅱ including TCN. Forecasting using two phases enable the model to sequentially learn the complex representative features of the original input data.

Phase Ⅰ extracts inter-relationships of MTS data with SACAE. Then, creates a condensed UTS representation of the input MTS data. During CAE process, the limitation of non-local dependency issue is mitigated by the self-attention inserted in the last and the first convolutional layer of the encoder and the decoder of CAE. The trained SACAE encoder passes the condensed UTS representation of MTS data to Phase Ⅱ. TCN, trained on intra-relationship of the data, then forecasts future time steps of the univariate representation time series. The forecasted time step is then decoded with the trained decoder of SACAE for MMTS operation and cause analysis. The structure of the proposed method is presented in <Figure 5>.

3.1 Phase Ⅰ

The proposed method utilizes SACAE to capture inter-relationship, the non-linear correlation existent in the input MTS data. SACAE encoder is consisted of multiple 1-D convolutional blocks and self-attention modules. Flatten layer is added in the fi-

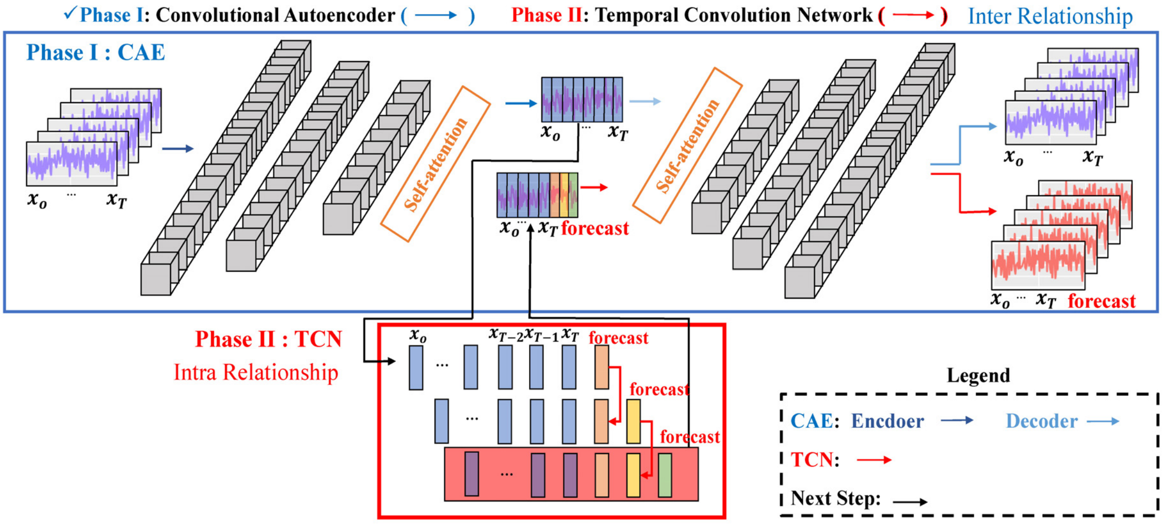

Figure 5. Overall Architecture of the Proposed Method

nal operation to flatten the contiguous range of dimensions into simple vector output.

MTS data is inputted into SACAE as channels while each 1-D convolution operation has kernel size of three, stride size of one, and padding size of one. By doing so, SACAE can preserve the overall length of the subsequent outputs after each convolutional block operations. The output which is the same length as the input enables TCN in Phase Ⅱ to forecast the next time step. This will be further discussed in the next section.

As data is passed along each convolutional block of SACAE encoder, the number of channels is exponentially decreased to extract abundant features of the original MTS input data. Each convolutional blocks of SACAE encoder decreases the number of channels of its input. Created channels then extract abundant information between individual variables of input MTS data. For example, data created from the same machinery of a manufacturing process. While CAE is able to capture the non-linear correlations of MTS data in the local field, the non-local dependency issue makes it difficult to extract the global dependencies. To mitigate such problem, self-attention is added to the last and the first layers of CAE. Now, SACAE calculates the importance of elements in the feature map and captures the global dependencies. We include self-attention in the deeper layers of convolutional blocks to capture global information as shallow convolutional layers are prone to capture local features of the input.

3.2 Phase Ⅱ

TCN is composed of causal convolutions and dilated convolutions. Causal convolutions extract relevant historical information and forecasts the future output while dilated convolutions facilitates computational efficiency. In a previous study, TCN has been proven to outperform baseline architectures of sequence modeling tasks (Bai *et al.*, 2018). In the proposed method, a baseline TCN with few changes has been used. The role of TCN is to forecast the next time steps of the encoded UTS representation of the original MTS data. The module predicts the next time steps of the encoded data based on the multiple temporal convolutional layers. TCN is composed of several temporal block layers with number of hidden channels in each layer either 25 or 50. The structure of TCN is presented in <Figure 3>. We increase the dilation rate d exponentially with the rate of 2^i. Thus, according to the number of layers, the filter may cover the whole input sequence.

SACAE decoder reconstructs the forecasted time steps of the UTS representation data back to the original input data. Through analysis of self-attention block of the decoder, cause analysis of the effects of the input data to the forecasted outputs is possible. SACAE decoder is similar to the structure of SACAE encoder. Multiple convolution blocks and self-attention layers are embedded. While the overall structure of both encoder and decoder is similar, SACAE decoder utilizes transposed 1-D convolutions.

4. Experiments

In this section, the proposed method is compared with various time series forecasting baseline models to evaluate its performance. Following conventional setups, ARIMA from statistical method, SVR from machine learning method, and LSTM from deep learning method is used for prediction evaluation comparison. Three standard metrics: modified mean absolute error (MMAE), modified mean squared error (MMSE), and modified R-Squared (MR^2) is used to measure the adequacy of the test prediction. Cause analysis is also conducted to evaluate the ability of the proposed method to extract important time steps of input data for forecast.

4.1 Datasets

The proposed method was verified on three real world datasets, namely CNC (Computer Numerical Control) (Sun, 2018), C-MAPPS (Turbofan Engine Degradation Simulation) (Saxena *et al.*, 2008), and PMM (Predictive Maintenance Modelling dataset) (Zonta, 2020). CNC is milling machine experiment dataset in the System-level Manufacturing and Automation Research Testbed (SMART) at the University of Michigan. C-MAPPS is engine degradation simulation dataset collected by NASA. PMM is telemetry reading and error identification, maintenance, and failure dataset created by Microsoft. Data was collected from a CNC machine for variations of tool condition, feed rate, and clamping pressure. The data was collected from 18 experiments with 100*m/s* sampling rate from four motors in the CNC milling machine. The proposed method was experimented on six variables of motor *x* and *y* as variables from motor *z* and *spindle* showed no significant variance in data. The data from C-MAPPS is collected from simulation of commercial turbofan engine data. Sampling was done at 1 Hz and consist of 30 engine and flight condition parameters. The proposed method was experimented with FD002_RUL data on fourteen parameters with 249 train time length and 20 test time length. PMM data was constructed for predicting problems caused by component failures. The data consist of eight variables including error identification time points and component maintenance time points. Experiment was conducted on 234 train length and 20 test length data.

<Figure 6> illustrates the relationships between six variables of CNC dataset through a pair grid plot. The six variables exhibit non-linear correlation between each other. Variables such as *x1_currentfeedback*, *x1_outputvoltage*, and *y1_outputvoltage* exhibit clearer non-linear correlation with other variables in the dataset.

To illustrate that *x1_outputvoltage* and *y1_commandvelocity* show clear non-linear correlation, we fitted a fourth order polynomial function as shown in <Figure 7> (Left) and witnessed that two variables do indeed inherit a non-linear correlation. However, fitting a linear regression to two variables in <Figure 7> (Right) do not properly represent the relationship between two variables. Drawing a conclusion that two variables inherit a complex non-linear correlation.

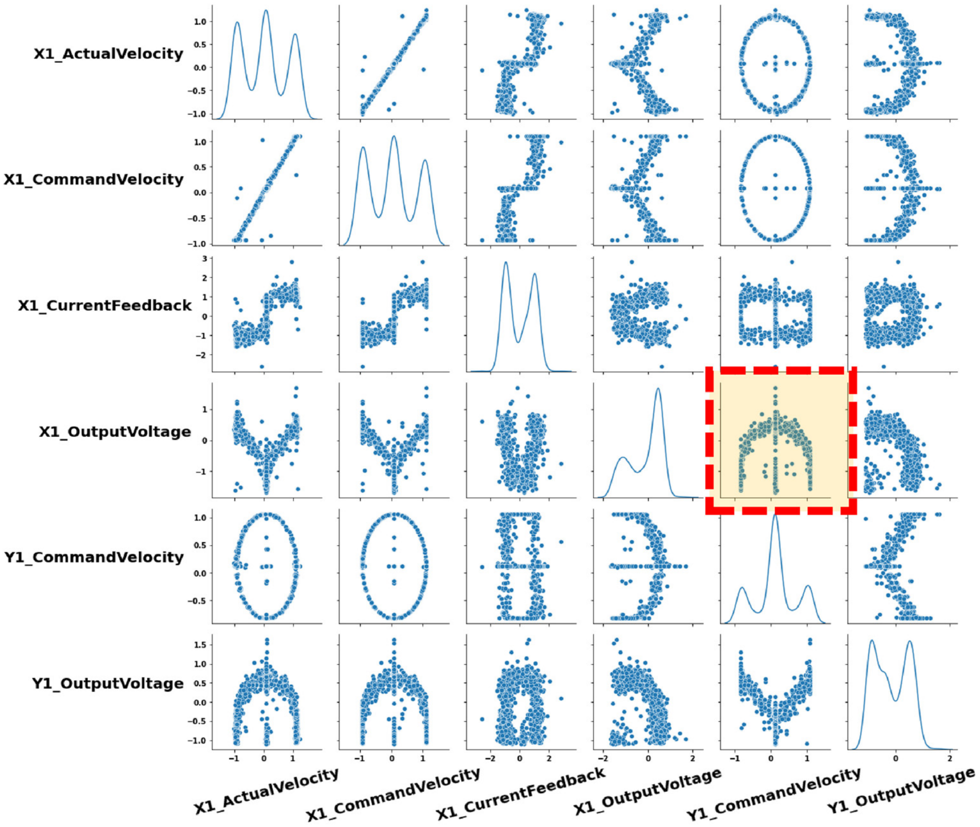

Figure 6. Non-linear Correlation Pair Grid of Six Variables in the Dataset and an Example of Explicit Non-linear Correlation of Variables *x1_outputvoltage* and *y1_commandvelocity*, Elaborated in <Figure 6>

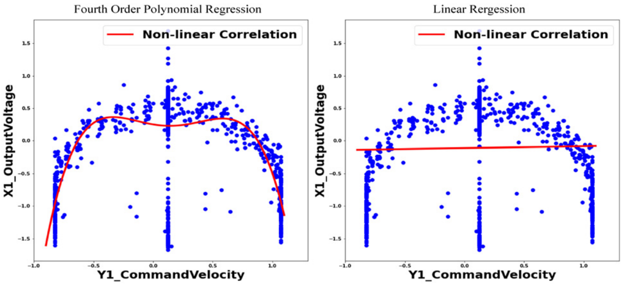

Figure 7. Fitted Image of Fourth order Polynomial Regression of variables *x1_outputvoltage* and *y1_commandvelocity* (Left) illustrates a clear non-linear correlation between two variables while linear regression (Right) does not properly represent the relationship between two variables

4.2 MMTS Forecast

In this study, we compare the proposed method with baseline sequence modeling benchmarks. Statistical, machine learning, and deep learning methods are compared with the proposed method. Models used to compare are ARIMA, SVR, and LSTM, baseline sequence models well known in each method. The results suggest that the proposed method outperforms the baseline sequence modeling architectures in capturing the high dimensional relationships of the variables.

Prediction accuracy was determined using MMAE, Equation (10), MMSE, Equation (11), and MR^2, Equation (12). These evaluation metrics are commonly used performance metrics for time series forecasting and are modified for MMTS forecasting.

$$MMAE = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{n}\Sigma_{t=1}^{n}\left|y_t^i - \hat{y}_t^i\right|\right) \tag{10}$$

$$MMSE = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{n}\sum_{t=1}^{n}(y_t^i - \hat{y}_t^i)^2\right) \tag{11}$$

$$MR2 = \frac{1}{m}\sum_{i=1}^{m}\left(1 - \frac{\Sigma(y_t^i - \hat{y}_t^i)^2}{\Sigma(y_t^i - \overline{y}^i)^2}\right) \tag{12}$$

n is the number of samples in the test set and m is the number of variables in the test set, $\overline{y}^i$ is the mean y^i of values, y_t^i is the actual output variable for i^{th} variable at time t and $\hat{y}_t^i$ is the predicted output for i^{th} variable at time t.

Shown in <Table 1>, the proposed method shows superior forecast capability and generally achieves better performance compared to other forecast baseline models. As expected, the baseline deep learning model performs similar to the baseline statistical model. We expect that the performance difference will exponentially increase with the increase in volume of data.

The proposed method exhibits strong performance improvement in C-MAPPS dataset. We conclude this is because of the ability of the proposed method to effectively learn the abundant non-linear correlations existent in C-MAPPS dataset. <Figure 8> illustrates the overall correlation between variables of C-MAPPS and PMM datasets. C-MAPPS with fourteen variables, except for sensor 15, show relatively high correlation values to other variables. On the contrary, PMM datasets, with about half the number of variables to C-MAPPS, do not show clearer correlations within the dataset.

The major advantage of the proposed method is its ability to learn inter-relationships, the non-linear relationship inherent in high-dimensional MTS data.

We now discuss the results of each model on forecasting individual variables to compare the ability of the proposed method to reflect the non-linear correlations. These results are presented in <Table 2>.

Prediction accuracy was determined by using mean square error (MSE), Equation (13). Compared to MMSE evaluation metric, MSE measures the prediction loss of individual variables in CNC dataset.

$$MSE = \frac{1}{n}\Sigma_{t=1}^{n}(y_t - \hat{y}_t)^2 \tag{13}$$

n is the number of samples in the test set, y_t is the actual output variable and $\hat{y}_t$ is the predicted output variable at time *t*.

<Table 2> represents MSE scores of the individual variables of CNC dataset forecast results of models. As shown in <Figure 6>, three variables *x1_currentfeedback*, *x1_outputvoltage*, and *y1_outputvoltage* of CNC dataset show strong explicit non-linear correlations with

Table 1. Forecasting Performance of Baseline Models and the Proposed Method

Mothod	CNC			C-MAPPS			PMM		
	MAE	MSE	R^2	MAE	MSE	R^2	MAE	MSE	R^2
ARIMA	0.201	0.132	0.783	1.12	1.79	0.002	0.643	0.936	0.438
SVR	0.169	0.086	0.791	0.199	0.086	0.932	0.772	1.543	0.183
LSTM	0.195	0.121	0.779	1.33	2.81	0.013	0.631	0.924	0.434
Proposed Model	**0.157**	**0.078**	**0.852**	**0.13**	**0.031**	**0.979**	**0.599**	**0.827**	**0.543**

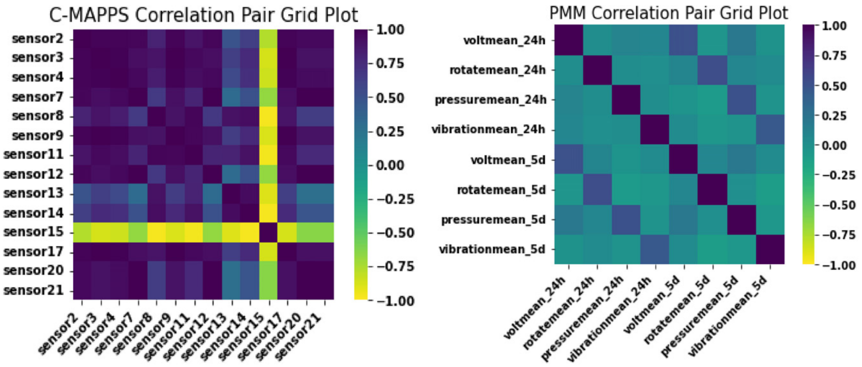

Figure 8. Pearson Correlation Plot of C-MAPPS (left) and PMM (right) Dataset

Table 2. Individual Mean Squared Error (MSE) Score on CNC Dataset Variables to Evaluate Models on Reflecting Non-linear Correlation

	x1_actualvelocity	*x1_commandvelocity*	*x1_currentfeedback*	*x1_outputvoltage*	*y1_commandvelocity*	*y1_outputvoltage*
ARIMA	**0.003**	**0.002**	0.223	0.223	0.223	0.116
SVR	0.001	0.017	0.110	0.193	0.051	0.128
LSTM	0.025	0.023	0.256	0.256	**0.025**	0.135
Proposed method	0.012	0.010	**0.105**	**0.180**	0.048	**0.114**

other variables. As an example, variables *x1_outputvoltage* and *y1_commandvelocity* has a complex correlation of a combination of linearity and non-linearity. Accordingly, the proposed method shows lowest MSE scores on all corresponding three variables. Also, shows the lowest MSE score on variable *x1_currentfeedback*, which has the clearest non-linear correlations amongst the six variables. The result of this experiment and the result of C-MAPPS dataset can conclude that the proposed method has strength in learning the non-linear correlations in MTS data, whereas other baseline models do not.

4.3 Cause Analysis

Besides forecasting MMTS, the proposed method also provides insights for cause analysis. For domain such as Prognostics and Health Management (PHM) in manufacturing, cause analysis is equally as important as forecasting future values. In real world applications,

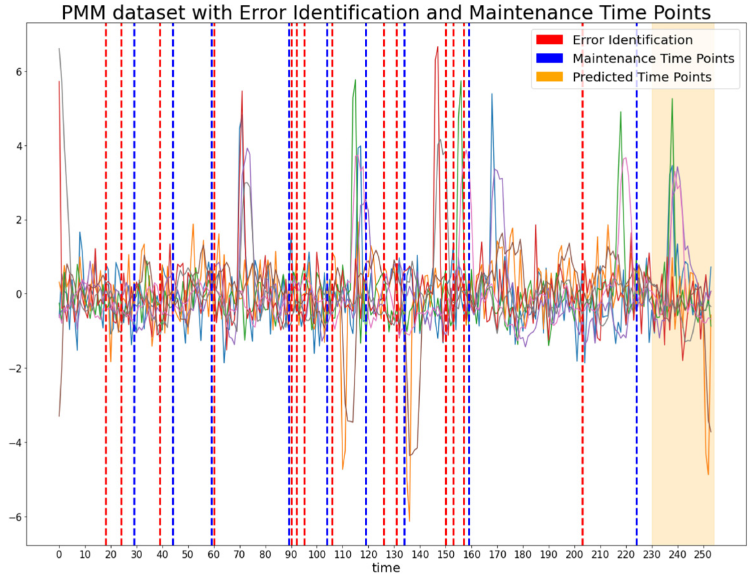

Figure 9. Visualization of PMM Original Dataset with Error Identification (red), Maintenance Time Points (blue) and Predicted 20 Time Points (orange)

cause analysis of MTS can aid engineers to discover root cause of accidents and save time and effort.

The self-attention map of SACAE decoder is used for cause analysis. The self-attention map mitigates the effect of long-term dependency problem and creates weights that help a model focus on important time steps in input data. It takes i number of inputs and outputs i number of outputs. The inputs will interact with each other to decide which time point is important and needs more attention. In our case, the self-attention focuses on time points that are crucial in forecasting the last 20 degradation points until failure of PMM dataset.

As shown in <Figure 9>, PMM dataset contains error identification and maintenance time points. These time points indicate the points in input data where error has been identified and the points where major component maintenances have been required. There are fifteen error identification time points and nine component maintenance time points. <Figure 9> shows visualization of original input data of PMM dataset with error identification, maintenance time points, and predicted time points.

The proposed method's diagnostic performance on PMM dataset is demonstrated on <Figure 10>. The self-attention module in SACAE decoder is used for cause analysis map. The cause analysis map indicates the time points the proposed model references strongly to forecast the last degradation time points leading up to the major failure time point. The row of cause analysis map indicates the last 20 degradation time points, and the column indicates the input time points from start to up to 234 time points. The darker columns of the cause analysis map indicate time points the self-attention module judged is important for forecasting the last 20 degradation time points.

As shown in <Figure 10>, the cause analysis map and the zoom-in version demonstrate the ability to locate crucial points in the time map of the input data. The red dotted lines indicate the error identification points, and the blue dotted lines indicate the maintenance time points of the input data. The maintenance time points are considered more crucial time points than the error identification time points as major defect has occurred which led to actual maintenance actions. The zoom-in version shows the self-attention map from 85 to 110 time points. All 20 prediction times focus strongly on time points 89 to 91, time points 93 to 95, and prediction time points 16 and 17 also focus on 103 to 105 time points. This indicates the model captures important time points of the input data to forecast the last degradation time points of PMM dataset. The cause analysis map has captured most major events of both error identification and maintenance time points. Especially, 7 out of 9 maintenance time points have been captured by the cause analysis map.

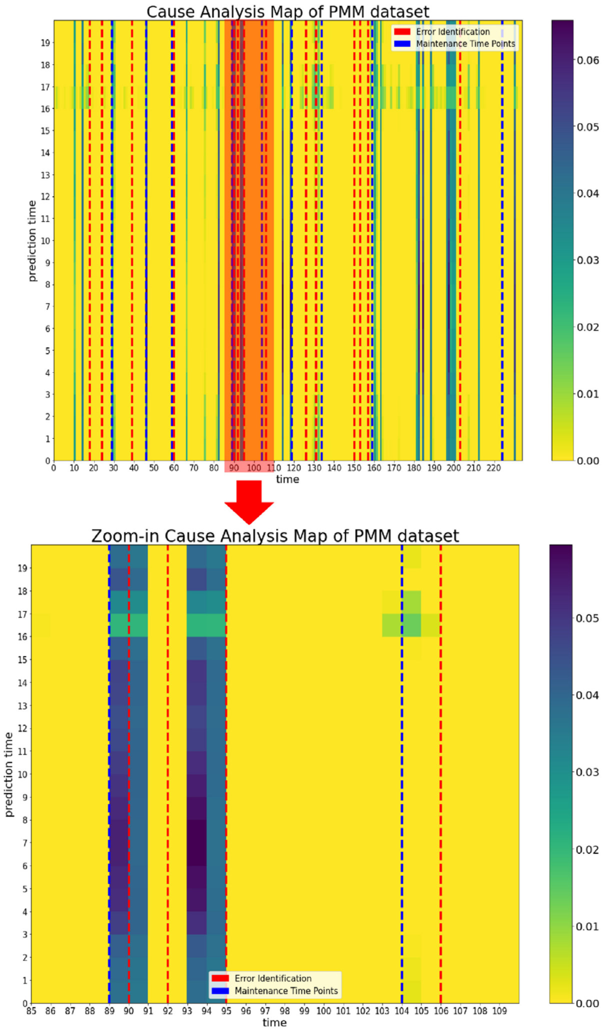

Figure 10. Original Cause Analysis and Zoom-in Cause Analysis Map of PMM dataset. Zoom-in Cause Analysis Map shows 85 to 110 time points of the Original Cause Analysis Map. The red and blue dotted lines indicate major events in the machine history. The cause analysis map of SACAE decoder focuses on these events or time points leading up to these events

5. Conclusion

In this paper, we proposed a two phase MMTS forecasting using SACAE and TCN. The model adopts a two phase process where in Phase Ⅰ, self-attention and convolutional autoencoder are used to create a univariate representation of the MTS data. This univariate representation contains inter-relationships of MTS data. In Phase Ⅱ, TCN is used to forecast the next time steps based on the compressed univariate representation. Finally, the forecasted time steps are decoded by the trained decoder of SACAE to reconstruct into the future time steps of the original individual variables.

High dimensional time series data are created in manufacturing domains. It is essential to forecast individual variables as these values can be utilized to help forecast demands, provide decisive statistics to initiate timely maintenance, or provide specific monitoring statistics. However, current baseline models cannot accurately reflect the non-linear correlations among variables in the MTS data for MMTS forecasting. Compared to these methods, the proposed method is more efficient as its simple architecture utilizes methods that minimizes computation cost. E.g., the convolutional operations in SACAE, the 1×1 convolutional operations in the self-attention module, and the dilated convolutional operations in TCN. Also, we have proven that the proposed method has strength in capturing complex inter-relationships of MTS data. The results show the proposed method exhibits better forecasting of MMTS ability. As for cause analysis results, the proposed model is tested on its ability to find causal time steps leading up to the final degradation operations. Finally, an end-to-end model, the proposed method is unique in that it operates as a MMTS forecasting model.

We have observed possibilities that varying CAE with differing improvements to the architecture may lead to improved performance. We plan to apply different methods to the module and conduct further experiments with various dataset as well. We expect the proposed method can be used with varying types of MTS data, outside the manufacturing domains as well.

References

Bai, S., Kolter, J. Z., and Koltun, V. (2018), An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, arXiv preprint arXiv:1803.01271.

Braei, M. and Wagner, S. (2020), Anomaly detection in univariate time-series: A survey on the state-of-the-art. arXiv preprintarXiv: 2004.00433.

Cerqueira, V., Torgo, L., and Soares, C. (2019), Machine learning vs statistical methods for time series forecasting: Size matters, arXiv preprint arXiv:1909.13316.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014), Empirical evaluation of gated recurrent neural networks on sequence modeling, arXiv preprint arXiv:1412.3555.

DiPietro, R. and Hager, G. D. (2020), Deep learning: RNNs and LSTM, In *Handbook of medical image computing and computer assisted intervention*, Academic Press, 503-519.

He, K., Zhang, X., Ren, S., and Sun, J. (2015), Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, In *Proceedings of the IEEE International Conference on Computer Vision*, 1026-1034.

Hyndman, R. J. and Athanasopoulos, G. (2018), *Forecasting: principles and practice*, OTexts.

Mawson, V. J. and Hughes, B. R. (2020), Deep learning techniques for energy forecasting and condition monitoring in the manufacturing sector, *Energy and Buildings*, **217**, 109966.

Meyes, R., Donauer, J., Schmeing, A., and Meisen, T. (2019), A recurrent neural network architecture for failure prediction in deep drawing sensory time series data, *Procedia Manufacturing*, 34, 789-797.

Morariu, C. and Borangiu, T. (2018, May), Time series forecasting for dynamic scheduling of manufacturing processes, In *2018 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, IEEE, 1-6.

Oord, A. V. D., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016), Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499.

Saxena, A., Goebel, K., Simon, D., and Eklund, N. (2008, October), Damage propagation modeling for aircraft engine run-to-failure simulation, In *2008 International Conference on Prognostics and Health Management*, IEEE, 1-9.

Sun, S., CNC Mill Tool Wear (CNC), Kaggle, 2018, https://www.kaggle.com/shasun/tool-wear-detection-in-cnc-mill/metadata.

Wang, J., Yan, J., Li, C., Gao, R. X., and Zhao, R. (2019), Deep heterogeneous GRU model for predictive analytics in smart manufacturing: Application to tool wear prediction, *Computers in Industry*, **111**, 1-14.

Wang, X., Girshick, R., Gupta, A., and He, K. (2018), Non-local neural networks, In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7794-7803.

Wu, P., Gong, S., Pan, K., Qiu, F., Feng, W., and Pain, C. (2021), Reduced order model using convolutional auto-encoder with self-attention, *Physics of Fluids*, **33**(7), 077107.

Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., and Zhang, C. (2020, August), Connecting the dots: Multivariate time series forecasting with graph neural networks, In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 753-763.

Yan, B. and Han, G. (2018), Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system, *IEEE Access*, **6**, 41238-41248.

Zhao, B., Lu, H., Chen, S., Liu, J., and Wu, D. (2017), Convolutional neural networks for time series classification, *Journal of Systems Engineering and Electronics*, **28**(1), 162-169.

Zheng Y., Q. Liu, E. Chen, Ge, Y., and Zhao, J. L. (2014), Time series classification using multi-channels deep convolutional neural networks, *Proc. of the 15th International Conference on Web-Age Information Management*, 298-
310.

Zonta, T. (2020), Predictive Useful Life based telemetry (PMM), Kaggle, https://www.kaggle.com/tiagotgoz/predictive-useful-life-based-into-telemetry/metadata.

Author Profile

Woo Young Hwang: received B.S. degree in Technological Systems Management from State University of New York Stony Brook University in 2020 and is currently working to get M.S. degree in industrial and management engineering form Korea University. His research interests include time series representation learning and prognostics and health management in manufacturing. He is conducting research to improve manufacturing systems using the latest deep learning technologies.

Jun-Geol Baek : received B.S., M.S., and Ph.D. degrees in industrial engineering from Korea University in 1993, 1995, and 2001, respectively. From 2002 to 2007, he was an assistant professor in the Department of Industrial Systems Engineering at Induk University, Seoul, Korea. He was also an assistant professor in the Department of Business Administration at Kwangwoon University, Seoul, Korea, from 2007 to 2008. In 2008, he joined the School of Industrial and Management Engineering, Korea University, where he is currently a professor. His research interests include fault detection and classification (FDC), advanced process control (APC), prognostics and health management (PHM), and big data analytics in manufacturing.